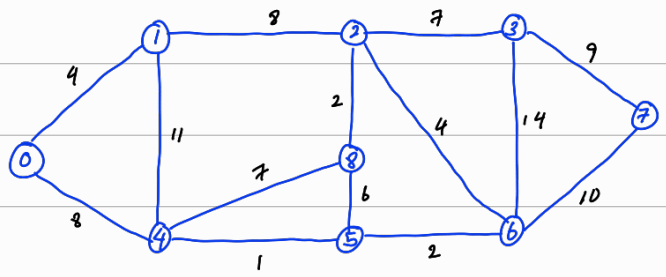
Dijksta Algorithm

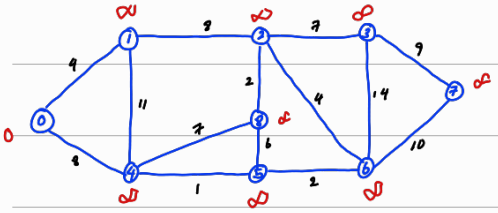


* This is use to Single source shortest path problem

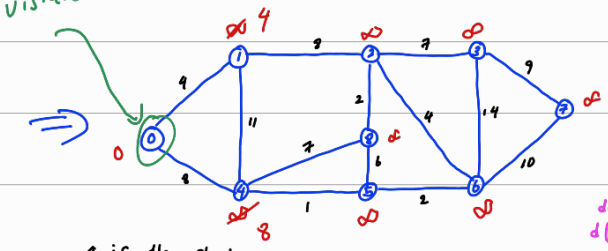
* In this case we Consider the 0 is the Source vertex.
[we can take any vertex as source vertex]

note: If node is visited then it not update again

using Condition → if $(d(v) + c(v,u) < d(u)$
$d(u) = d(v) + c(v,u)$



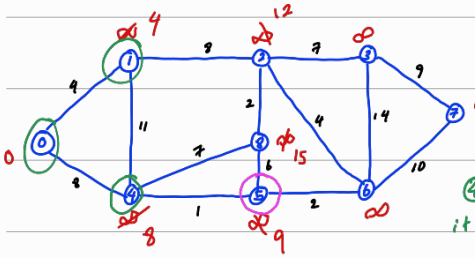
visited



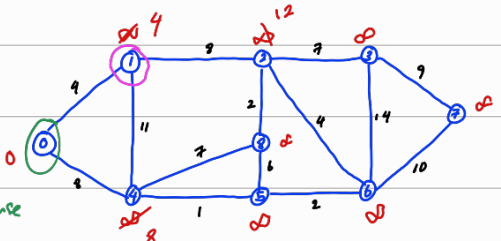
4 is the shortest one therefore go to ①

node ④
$d(v) = 0$
$d(u,v) = 8$
$d(w) = \infty$

node ①
$d(v) = 0$
$d(u,v) = 4$
$d(w) = \infty$

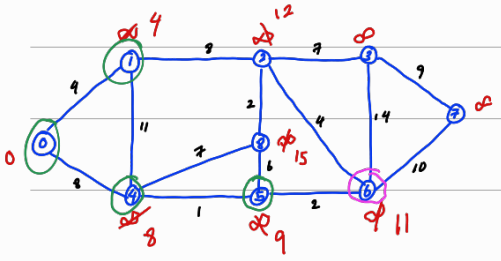


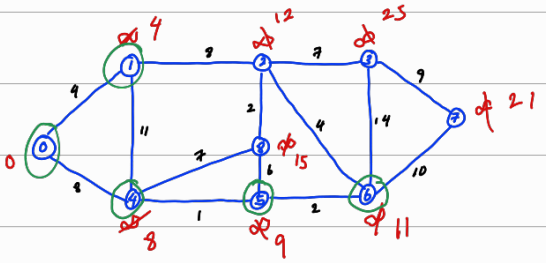
we select the ④ node because it has shortest distance



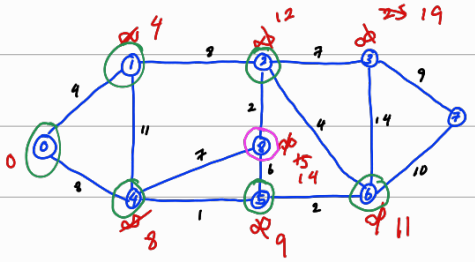
* when consider the node ④ it's distance is 15 related to ① node but it current distance is 8 therefor it not update

* next we find the shortest distance node that not visited

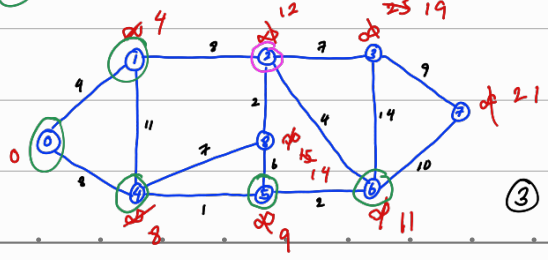




next goto ②



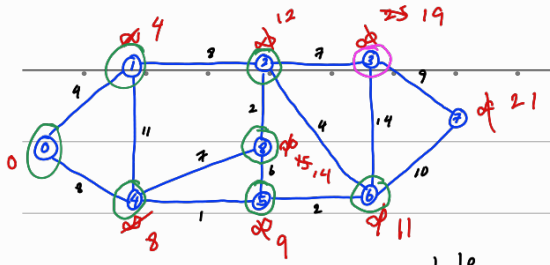
next



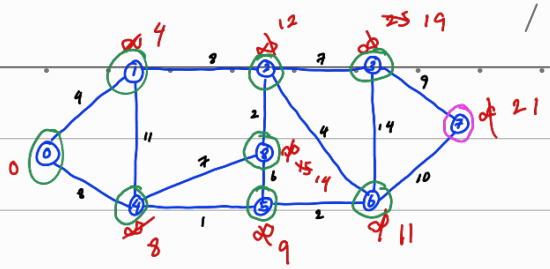
③ and ⑧ updated.

nothing to update

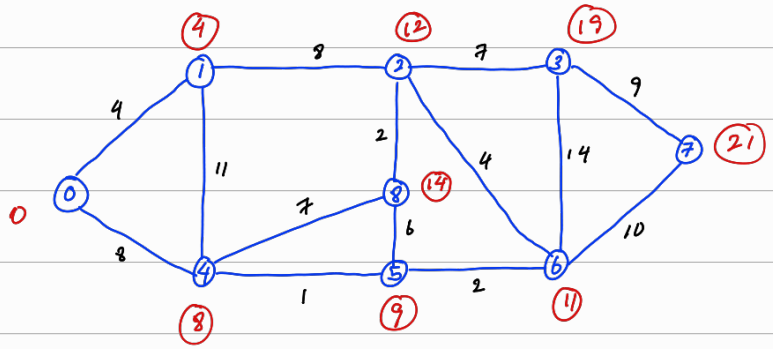
next (3)



nothing to update

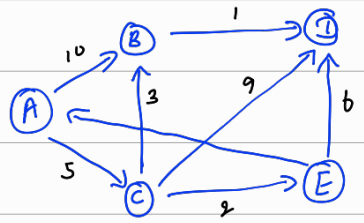


Now visited do the all nodes.



* This algorithm working on undirected & directed graphs.

ex: Let's see the directed graph



* Suppose (A) is the source vertex

source vertex is must be first

visited vertices	A	B	C	D	E
A	0	∞	∞	∞	∞
C		10	5	∞	∞
E		8		14	7
B		8		13	
D				9	

what is the shortest distance of

A to D = 9

A to E = 7

A to C = 5

path of A to D

visited vertices	A	B	C	D	E
A	0 ←	∞	∞ ←	∞	∞
C		10 ←	5 ←	∞	∞
E		8 ←		14	7
B		8 ← then		13 changed ←	
D				9 ←	

first point to the D and go to the previous row and check D is changed or not if it is changed then go to the pointer to that row selected node. and check the previous row and do some thing

path = D B C A

= A C B D = 9

A to B

visited vertices	A	B	C	D	E
A	0 ←	∞	∞	∞	∞
C		10 ←	5 ←	∞	∞
E		8 ←		14	7
B		8 ←		13	
D				9	

path = B C A

= A C B = 8